Supporting the SBR Style of Web Usage

Boworn Leemakul, Panyapon Saeliw, and Andrew Davison
Department of Computer Engineering
Prince of Songkla University
Hat Yai, Songkhla 90112, Thailand

Abstract

The Search-Browse-Repeat (SBR) mode of Web usage is becoming increasingly important as the size of the Web increases and the coverage of search engines expands. However, Web browsers do little to support SBR, with consequences for search success rates. We characterise the main features of SBR, suggest criteria that would help alleviate the problems associated with it, and outline a prototype browser we are building which embody these ideas.

Keywords: *web-based computing, browser navigation, visualisation*

1. Introduction

The current size of the Web is probably between 3 and 4 billion pages [2], and so it is hardly surprising that search engines are so popular: Google (`http://www.google.com`) alone estimates that it receives about 130 million queries per day [13]. Unfortunately, search engines are not keeping up with Web expansion: Google indexes about 1.5 billion pages, with the other popular engines far behind [12]. This may partly explain why the failure rate for searches is estimated at nearly 20% [11], and why users find that receiving a search engine's results page is only the beginning of their search.

In section 2, we describe the Search-Browse-Repeat (SBR) mode of Web search, which characterises a user's experience when searching for a page from the starting point of a search engine query. Many of the problems that arise during SBR are poorly handled by today's Web browsers. In section 3, we outline our criteria for improved browser support of SBR, and in section 4 we describe a prototype that follows these guidelines. Section 5 discusses some of the unresolved issues raised by our work.

2. Characterising SBR

The description of SBR in this section is based on anecdotal evidence: observations made during several training courses teaching novices how to use the Web, and on our own experiences.

SBR is a search activity that begins with the user *sending a query to a search engine*. There are other forms of search, such as a search of a large organisation's Web site by browsing from their home page (perhaps with the help of their site map), or a search of a portal site such as Yahoo. These kinds of search are different from SBR in that the search domain (i.e. Web site, portal) has been organised into a coherent form with the Web equivalent of maps and signposts. SBR lacks this structuring, which is the underlying cause of user problems.

The search engine query which begins SBR returns a Web page of matching results, often numbering in the many 1000's. There is rarely any apparent ordering to the results, and the sheer number can easily cause the 'right' link to be missed.

A sophisticated user will try to reformulate the query to reduce the number of hits, but novices do not have the skills to

do this. Instead, they resort to manually investigating each link, which usually means a very rapid visit to the page (perhaps spending only a few seconds there) before hitting the back button. Novices are unwilling to read the text fragments that accompany the links returned by the search engine.

A typical link will lead to a page deep inside an organisation's site, at some distance from a table of contents or site map. Also, pages written by individuals frequently contain little navigational support. The searcher must employ browsing to move from the link page to a page holding the information they need. This information is usually located at the same site, often just 1-2 links away.

Browsing, often without much contextual information apart from anchor text, leads to inadvertent jumps to distant pages, and a growing sense of disorientation as the page trail increases. For example, novice users will frequently click on banner ads or miss links embedded inside image maps. They can even forget the aim of their original search, as they become more distracted. Browsing behaviours of this kind mean that long trails do not have a single logical meaning (e.g. 'find a page about subject 'X').

The back button becomes very important in order for the user to 'get out' of useless pages. However, its stack-based nature means that trails (both good and bad) are lost as the user backtracks [7]. Other browser navigation aids (e.g. the history list, bookmarks) are rarely used.

A problem unique to our students is that English is not their first language, which compounds their problems when rapidly scanning Web pages and reading anchors.

Many users will simply 'start over' after a certain amount of time, and send exactly the same query to the search engine again. There is some small evolution of the user's search strategy (e.g. new keywords), but most users admit to have becoming so confused during the previous search as to be unsure how to refine their query.

3. Improving SBR Support in Browsers

We now describe broad criteria for improving SBR support in Web browsers. Our discussion follows the categories outlined in [5].

A visualisation of the search space would be of enormous benefit to the user. Due to the nature of SBR, it must be generated dynamically and be a partial view of the space. There are two principle 'schools' of Web visualisation: trails of jumps between pages and maps based on the organisation of a site.

As mentioned above, a long trail begins to lose meaning as the user get distracted or changes their search aims. Over time, the amount of nonrelevant pages and links will swamp the useful information. More complex heuristics to determine a meaningful trail might help, such as dwell time and referrer consistency as used in the Footprints system [15].

Site maps (e.g. contents lists, tables, frames) are usually statically created, and applied to a single coherent Web entity (e.g. a business' site) [1].

A computational viable dynamic map cannot survey an entire site: it must display partial information. Also, it is generally impossible to analyse the logical relationships between pages, but a hierarchy based on URL structure is simple to create. This results in a *sites tree*, where the path of a URL becomes nested 'folders' which ultimately contain a node representing the URL filename. URLs located at different sites create distinct branches at the top-level of the tree.

The advantage of this view is its great familiarity to novice users from applications like Windows Explorer. The disadvantage is that the hierarchy is geographical rather than relational. However, for SBR a geographical display is very helpful: often the user will only want to navigate to other pages within the same site in order to find relevant information. Also, an Explorer-like interface is suited to displaying 100's of

nodes, and has a view mechanism based on expanding and closing folders.

Another criteria is whether to employ 2D or 3D visualisation. 2D visuals are cheap to create, update, and rearrange for a reasonable number of nodes. 3D is generally expensive, and there are concerns about problems such as occlusion and its suitability for displaying large amounts of text [6].

Should the Web be represented by a graph or a tree? While a graph is the more natural model for Web interconnectivity, as a visualisation model it soon becomes cluttered, hard to understand and navigate. Complex graphs can be costly to generate and redraw when the user's point of view moves. A tree is simpler to construct, but has less flexible relationships. However, it is a good choice for our sites tree, which is overridingly hierarchical.

Filtering of the visualisation is essential so that extraneous detail can be hidden. The tree model has a familiar visual filtering model based on folder opening and closing. Further semantic filtering is necessary, perhaps based on the content of pages or link meaning. Utilising information based only on titles can cause problems due to missing/wrong titles or poorly named pages [3]. Also, we wish to avoid query languages for filtering since they seem too complex for novices.

There should be a predictive element to the visualisation, to guide browsing from the current page location. It must be simple for novices to understand, and not be prohibitive to calculate dynamically.

4. A SBR Browser Prototype

Figure 1 shows our prototype SBR Browser in operation. The top row contains a field for downloading a URL, and a search button which sends a query to Google. The central part of the browser is divided into three columns: the left column is the sites tree display, the middle area shows the Web page, and the right-hand

Figure 1. The SBR Browser Prototype.

column holds a pop-down list of links, a page summary window, and a score area. The browser is coded in Java.

The prototype is best explained by considering its contribution to the three phases of SBR.

The Search Phase. A search query is sent to Google, and the results page is shown back in the browser. The links in the results page are automatically extracted and their Web pages downloaded in the background (i.e. they are not displayed). As a page arrives, it is summarised, and scored. The summary is derived from the words on the page, excluding stop words and HTML tag labels. Scoring is a simple calculation which judges how similar the summary words are to the search query keywords. The URLs of the retrieved pages are added to the sites tree.

Each URL is represented by a series of nested folders corresponding to its path, with a file node for the URL filename. The node shows the name of the URL and its score. Right clicking on the node displays the page title and the summary words. Scores are propagated up through the folders to the top level of the tree. If two URLs share a common path, then the higher of the two scores are passed upwards.

The sites tree closes all of its folders apart from the path to the node with the highest score, and the path to the current page in the display window.

In the right hand area, the pop-down list contains the URLs of the links and their scores. The list is sorted into decreasing order by score.

The Browsing Phase. The scores in the sites tree guide the user towards the most promising page to examine. A page can be downloaded and shown either by clicking on a node in the sites tree, the pop-down list, or a link in the current Web page.

The newly retrieved page's links are displayed in the pop-down list, and a summary of the page appears in the "Page Summary" window. This window can either be set to show a summary of the entire page, or summaries by section. The score for the page also appears.

If the page is considered relevant then the user can click on the "Summarize Links" button. This causes all the pages linked to the current page to be downloaded in the background, summarised, and scored. This information is added to the sites tree as new nodes. If the highest scoring node changes, then the folders leading to the new node are opened.

The user-controlled "Summarize Links" button is a compromise for efficiency. The retrieval of all the links is a costly activity, and so we chose not to automate it.

The Repeat Phase. The user can refine a search in two ways. The keywords can be adjusted in the search keywords field, and the current nodes in the sites tree can be rescored. This involves no network communication, so it a fast operation. The choice of keywords is assisted by examining the page summaries. The other approach is to send a new query to Google, which will cause the old sites tree to be discarded, and a new one initiated.

5. Discussion

Search-Browse-Repeat (SBR) is characterised by a search engine query returning numerous links to disparate pages, followed by substantial browsing activity to find information. The browsing is typified by a lack of contextual information, long trails, large numbers of choice points in the search, extensive backtracking, and the problems of distraction and inadvertent jumps to distant points. Browsing often ends with a repeat phase where the user searches again, sometimes with a refined query.

Our criteria for supporting SBR in Web browsers are to utilise dynamic visualisation of partial maps of the search sites, represented as a 2D sites tree. The top-level branches of the tree are distinct Web sites, and the branches below represent the URL paths. Filtering utilises a mixture of standard visual tree techniques, and semantic notions based

around page summaries and scores. The scores are used as a predictive element to guide browsing.

Preliminary tests of the prototype show that training time is necessary for users to understand the SBR approach. Once this has been mastered, results can sometimes be found very quickly. However, this is heavily dependent on the scoring function which has proved to be unreliable. The summaries are helpful in query refinement, but are frequently too simplistic.

A fundamental question about SBR is whether it really is as pervasive as we believe. Our views on SBR are based on a sample size of about 50 people who were novice Web users, and attending a course aimed at learning search techniques (amongst other things). We are unaware of any study on this matter apart from [14], which reported that the submission of forms data (e.g. for search engine queries) accounted for only 4% of a user's navigation activities. However, search engine capabilities have changed enormously since 1996-1997, and a new study of usage patterns should be undertaken.

Our prototype shows that summarising and scoring operations are crucial. We are in the process of replacing our original code with better indexing, Porter stemming [10], and scoring (based on the Lucene package [9]).

A stubborn problem is the network load caused by the analysis of all the pages linked to the current page. Our code only downloads the text of these pages, but this is still quite slow (especially with the bandwidth available to us). The number of links is the crucial variable; our tests have uncovered pages with 50+ links.

An interesting visualisation technique we are currently considering is interaction histories [8], where annotations are used to signal the paths already investigated. This could be something as simple as changing the colour of nodes which have already been examined. Also of interest is the interface in Webview for representing cross-site navigational links as arrows [4].

References

[1] Brunk, B. 1999. "Overview and Preview Tools for Navigating the World-Wide Web", *SILS Technical Report* TR-1999-03, DoCS, Univ. of North Carolina at Chapel Hill, July.

[2] Client Help Desk. 2000. "Web Statistics: Size, the Average Page", Available at `http://www.clienthelpdesk.com/statistics_research/web_statistics.html`, July.

[3] Cockburn, A. and Greenberg, S. 1999. "Issues of Page Representation and Organization in Web Browser's Revisitation Tools", *Proc. OzCHI'99*, Wagga Wagga, Australia, pp.7-14, November.

[4] Cockburn, A., Greenberg, S., McKenzie, B., Smith, M., and Kaasten, S. 1999. "Webview: A Graphical Aid for Revisting Web Pages", *Proc. OzCHI'99*, Wagga Wagga, Australia, pp.15-22, November.

[5] Cockburn, A. and Jones, 1997. "Design Issues for World Wide Web Navigation Visualisation Tools", *Proc. of RIAO'97*, Montreal, Canada, p.55-74, June.

[6] Cockburn, A. and McKenzie, B. 2000. "An Evaluation of Cone Trees", In *People and Computers XV* (Proc. of the 2000 British Computer Society Conf. on Human-Computer Interaction), Sunderland, UK, pp.425-436.

[7] Greenberg, S. and Cockburn, A. 1999. "Getting Back to Back: Alternate Behaviours for a Web Browser's Back Button", *Proc. of the 5th Annual Human Factors and the Web Conf.*, Gaithersburg, Maryland, USA, June.

[8] Hill, W. C., Hollan, J. D., Wroblewski, D., and McCandless, T. 1992. "Edit Wear and Read Wear",

Proc. of CHI'92 Conf. on Human Factors in Computing Systems, pp.3-9.

[9] Jakarta Lucene. 2002. "Lucene", *Apache Jakarta Project*. Available at `http://jakarta.apache.org/lucene/docs/index.html`

[10] Porter, M.F., 1980, "An Algorithm for Suffix Stripping", *Program*, 14(3), pp.130-137. Code available at `http://www.tartarus.org/~martin/PorterStemmer/`

[11] Search Engine Watch 2000. "NPD Search and Portal Site Study", *Search Engine Watch*, Available at `http://searchenginewatch.com/reports/npd.html`, July.

[12] Search Engine Watch 2002a. "Search Engine Sizes", *Search Engine Watch*, Available at `http://searchenginewatch.com/reports/sizes.html`, January.

[13] Search Engine Watch 2002b. "Searches Per Day", *Search Engine Watch*, Available at `http://searchenginewatch.com/reports/perday.html`, February.

[14] Tauscher, L. and Greenberg, S. 1997. "How People Revisit Web Pages: Empirical Findings and Implications for the Design of History Systems", *Int. Journal of Human Computer Studies*, Special Issue on World Wide Web Usability, 47(1), pp.97-138.

[15] Wexelblat, A. and Maes, P. 1999. "Footprints: History-Rich Tools for Information Foraging", *Proc. of CHI'99*, Pittsburgh, USA, pp.270-277, May.